UNIVERSITY OF DORTMUND

REIHE COMPUTATIONAL INTELLIGENCE

COLLABORATIVE RESEARCH CENTER 531

Design and Management of Complex Technical Processes and Systems by means of Computational Intelligence Methods

Evolutionary Algorithms and the Maximum Matching Problem

Oliver Giel and Ingo Wegener

No. CI-142/02

Technical Report ISSN 1433-3325 December 2002

Secretary of the SFB 531 · University of Dortmund · Dept. of Computer Science/XI
44221 Dortmund · Germany

This work is a product of the Collaborative Research Center 531, "Computational Intelligence," at the University of Dortmund and was printed with financial support of the Deutsche Forschungsgemeinschaft.

Evolutionary Algorithms and the Maximum Matching Problem

Oliver Giel* and Ingo Wegener*

FB Informatik, LS 2, Univ. Dortmund, 44221 Dortmund, Germany
{giel, wegener}@ls2.cs.uni-dortmund.de

Abstract. Randomized search heuristics like evolutionary algorithms are mostly applied to problems whose structure is not completely known but also to combinatorial optimization problems. Practitioners report surprising successes but almost no results with theoretically well-founded analyses exist. Such an analysis is started in this paper for a fundamental evolutionary algorithm and the well-known maximum matching problem. It is proven that the evolutionary algorithm is a polynomial-time randomized approximation scheme (PRAS) for this optimization problem, although the algorithm does not employ the idea of augmenting paths. Moreover, for very simple graphs it is proved that the expected optimization time of the algorithm is polynomially bounded and bipartite graphs are constructed where this time grows exponentially.

1 Introduction

The design and analysis of problem-specific algorithms for combinatorial optimization problems is a well-studied subject. It is accepted that randomization is a powerful concept for theoretically and practically efficient problem-specific algorithms. Randomized search heuristics like random local search, tabu search, simulated annealing, and variants of evolutionary algorithms can be combined with problem-specific modules. The subject of this paper are general and not problem-specific search heuristics. Practitioners report surprisingly good results which they have obtained with such search heuristics. Nevertheless, one cannot doubt that problem-specific algorithms outperform general search heuristics – if they exist. So the area of applications of general search heuristics is limited to situations where good problem-specific algorithms are not known. This may happen if one quickly needs an algorithm for some subproblem in a large project and there are not enough resources (time, money, or experts) available to develop an efficient problem-specific algorithm. In many real-life applications, especially in engineering disciplines, there is no possibility to design a problem-specific algorithm. E. g., we may have the rough draft of a machine but we still have to choose between certain alternatives to obtain an explicit description of the machine. If we have m binary decisions to take, the search space (the space of all

* Supported in part by the Deutsche Forschungsgemeinschaft (DFG) as part of the Collaborative Research Center "Computational Intelligence" (SFB 531).

possible solutions) equals $\{0,1\}^m$. Then there exists a function $f\colon \{0,1\}^m \to \mathbb{R}$ such that $f(a)$ measures the quality of the machine if the vector of alternatives $a = (a_1, \dots, a_m)$ is chosen. However, often no closed form of f is known and we obtain $f(a)$ only by an experiment (or its simulation).

We conclude that general randomized search heuristics have applications and that their analysis is necessary to understand, improve, and teach them.

It is not possible to analyze algorithms on "unknown" functions f. However, one can improve the knowledge on a search heuristic by

- analyzing its behavior on some classes of functions,
- analyzing its behavior on some well-known combinatorial problems,
- constructing example functions showing special properties of the heuristic.

Such results have been obtained recently for evolutionary algorithms. Evolutionary algorithms have been analyzed on unimodal functions (Droste, Jansen, and Wegener (1998)), linear functions (Droste, Jansen, and Wegener (2002)), quadratic polynomials (Wegener and Witt (2002)), and monotone polynomials (Wegener (2001)). Among other properties the effect of crossover has been studied (Jansen and Wegener (2001, 2002)). A first step to study evolutionary algorithms on combinatorial problems has been made by Scharnow, Tinnefeld, and Wegener (2002) who studied sorting as minimization of unsortedness of a sequence and the shortest path problem. These problems allow improvements by local steps. Here, we investigate one of the best-known combinatorial optimization problems in P, namely the maximum matching problem.

We work with the following model of the problem. For graphs with n vertices and m edges, we have to decide for each edge whether we choose it. The search space is $\{0,1\}^m$ and a search point $a = (a_1, \dots, a_m)$ describes the choice of all edges e_i where $a_i = 1$. The function f to be optimized has the value $a_1 + \cdots + a_m$ (the number of edges) for all a describing matchings, i. e., edge sets where no two edges share a vertex. For all non-matchings a, the so-called fitness value $f(a)$ is $-c$ where the collision number c is the number of edge pairs e_i and e_j that are chosen by a and share a vertex. This definition is crucial. If we chose $f(a) = 0$ for all non-matchings a then our algorithm (and many other randomized search heuristics) would not find any matching in polynomial time, e. g., for the complete graph.

The maximum matching problem has the following nice properties:

- There is a well-known optimization strategy by Hopcroft and Karp (1973) which is based on non-local changes along augmenting paths,
- there are graphs where the Hamming distance between a second-best search point a and the only optimal search point is as large as possible (see Sect. 3), namely m,
- for each non-maximum matching a, there is a sequence $a^0 = a, a^1, \dots, a^\ell$ such that $f(a^0) = f(a^1) = \cdots = f(a^{\ell-1}) < f(a^\ell)$, Hamming distances $H(a^i, a^{i+1}) \le 2$, and $\ell \le \lceil n/2 \rceil$,
- and Sasaki and Hajek (1988) have investigated simulated annealing on it.

Simulated annealing only explores Hamming neighbors and, therefore, has to accept worse matchings from time to time. Evolutionary algorithms frequently consider new search points with larger Hamming distance to their current search point. We investigate a simple mutation-based evolutionary algorithm (EA) with population size one. Our conjecture is that larger populations and crossover do not help. The basic EA consists of an initialization step and an infinite loop. Special mutation operators will be introduced in the next paragraph.

Initialization: Choose $a \in \{0,1\}^m$ according to the uniform distribution.
Loop: Create a' from a by mutation and replace a by a' iff $f(a') \geq f(a)$.

In applications, we need a stopping criterion but typically we never know whether a is optimal. Hence, we are interested in X, the minimum t such that we obtain an optimal a in step t. This random variable X is called the optimization time of the algorithm. The standard mutation operator decides for each bit a_i of a independently whether it should be flipped (replaced by $1 - a_i$). The flipping probability equals $1/m$ implying that the expected number of flipping bits equals one. This algorithm is called (1+1) EA. We can compute $f(a)$ for the first a in time $O(m)$ and all successive a' in expected time $O(1)$ each (see Appendix A). Hence, $\mathrm{E}(X)$ is an approximative measure of the runtime. Since we have seen that steps with at most two flipping bits suffice to find an improvement, we also investigate the *local* (1+1) EA; in each step with probability 1/2 a randomly chosen bit a_i flips and with probability 1/2 a randomly chosen pair a_i and a_j flips. Sometimes it is easier to understand some ideas when discussing the local (1+1) EA. However, only the (1+1) EA is a general randomized search heuristic optimizing eventually each function $f\colon \{0,1\}^m \to \mathbb{R}$. In particular, the (1+1) EA (and also its local variant) does not employ the idea of augmenting paths and it is interesting to investigate whether it nevertheless randomly finds augmenting paths. Such a result would be a hint that evolutionary algorithms may implicitly use an optimization technique without knowing it. Again we stress that our aim is the investigation of evolutionary algorithms and we definitely do not hope to improve the best known maximum matching algorithms (Micali and Vazirani (1980), Vazirani (1994), Blum (1999)). Here, we mention that our model of the matching problem allows a polynomial-time algorithm even if the graph is not given explicitly and the algorithm only sees f-values (see Appendix B).

In Sect. 2, we show that the considered EAs always find matchings easily. It is proved that the EAs are polynomial-time randomized approximation schemes (PRAS) for optimization problems. This is a fundamental result, since approximation is the true aim of heuristics. In Sect. 3, we describe how the EAs work efficiently on paths and, in Sect. 4, we describe graphs where the EAs have an exponential expected optimization time.

2 Evolutionary Algorithms are PRAS

For many graphs it is very likely that the initial search point is a non-matching. However, the (local) (1+1) EA finds matchings quickly.

Lemma 1. *The (local) (1+1) EA discovers a matching in expected time* $O(m^2)$.

Proof. Assume that initially the collision number is $c > 0$, i. e., there exist c edge pairs $\{e_i, e_j\}$ such that e_i and e_j have an endpoint in common. Let t denote the total number of distinct edges contained in any of these edge pairs. Since t edges can form at most $\binom{t}{2} \le t^2$ edge pairs, $c \le t^2$ and $t \ge \sqrt{c}$ hold. For the (1+1) EA, the probability that exactly one of these t edges flips is $t(1/m)(1-1/m)^{m-1} \ge t/(em) \ge \sqrt{c}/(em)$, and for the local (1+1) EA, it is $t/(2m) \ge \sqrt{c}/(2m) \ge \sqrt{c}/(em)$. The expected time until the number of colliding pairs decreases by at least one is at most $(em)/\sqrt{c}$. Thus, the expected time to find a matching is upper bounded by the sum of expectations

$$em \sum_{c \ge i \ge 1} \frac{1}{\sqrt{i}} \;\le\; em \int_0^c \frac{1}{\sqrt{x}}\,dx \;=\; em\Big[2x^{1/2}\Big]_0^c \;=\; 2em\sqrt{c} \;=\; O(m^2). \qquad \square$$

Now we are prepared to prove that the (local) (1+1) EA efficiently finds at least almost optimal matchings.

Theorem 1. *For $\varepsilon > 0$, the (local) (1+1) EA finds a $(1+\varepsilon)$-approximation of a maximum matching in expected time $O(m^{2\lceil 1/\varepsilon \rceil})$.*

Proof. By Lemma 1, we can assume that the EA has found a matching M. If M is not optimal, there exists an augmenting path $e_{i(1)}, \dots, e_{i(\ell)}$, where ℓ is odd, $e_{i(j)} \notin M$ for j odd, $e_{i(j)} \in M$ for j even, and no edge in M meets the first or last vertex of the path. The (1+1) EA improves M by flipping exactly the edges of the augmenting path. This happens with probability $\Omega(m^{-\ell})$. The local (1+1) EA improves M by $\lfloor \ell/2 \rfloor$ 2-bit mutations shortening the augmenting path from left or right and a final 1-bit mutation changing the free edge of the resulting augmenting path of length one into a matching edge. The probability that this happens within the next $\lfloor \ell/2 \rfloor + 1$ steps is bounded below by $\Omega\big((m^{-2})^{\lfloor \ell/2 \rfloor} \cdot m^{-1}\big) = \Omega(m^{-\ell})$. If we can ensure that there always exists an augmenting path whose length is at most $\ell = 2\lceil \varepsilon^{-1} \rceil - 1$, the expected time to improve the matching is bounded by $O(m^{\ell})$ for the (1+1) EA and $O(\ell \cdot m^{\ell})$ for the local (1+1) EA. For ε a constant, $O(\ell \cdot m^{\ell})$=$O(m^{\ell})$. In fact, the bound $O(m^{\ell})$ for the local (1+1) EA holds for arbitrary $\varepsilon > 0$ (see Appendix C). Hence, for both EAs, the expected overall time is $O(m^2) + O(m) \cdot O(m^{\ell}) = O(m^{2\lceil \varepsilon^{-1} \rceil})$.

We can apply the known theory on the maximum matching problem to prove that bad matchings imply short augmenting paths. Let M^* be an arbitrary but fixed maximum matching. We assume $|M^*| > (1+\varepsilon)|M|$, i. e., the (1+1) EA has not yet produced a $(1+\varepsilon)$-approximation. Furthermore, let $|M| \ge 1$; otherwise there exists a path of length $1 \le \ell$. Consider the graph $G' = (V, E')$ with edge set $E' = M \oplus M^*$, where $\oplus$ denotes the symmetric difference. G' consists of paths and cycles, forming the components of G'. All cycles and all paths of even length consist of the same number of M-edges as M^*-edges, whereas paths of odd length have a surplus of one M^*-edge or one M-edge. That means, all paths of odd length starting with an M^*-edge also end with an M^*-edge and are augmenting paths relative to M. Let $k := |M^*| - |M|$. Then $|M^*| > (1+\varepsilon)|M|$ implies $k/|M| > \varepsilon$. There exist at least k disjoint paths of the last kind and at least one of them has no more than $\lfloor |M|/k \rfloor \le \lfloor \varepsilon^{-1} \rfloor$ M-edges. In fact, if ε^{-1} is an integer, then $|M|/k < \varepsilon^{-1}$ implies $\lfloor |M|/k \rfloor < \lfloor \varepsilon^{-1} \rfloor$. Thus the path has at most $\lceil \varepsilon^{-1} \rceil - 1$ M-edges and its total length is at most $\ell = 2\lceil \varepsilon^{-1} \rceil - 1$. $\square$

The next corollary is an easy application of Markov's inequality.

Corollary 1. *According to Theorem 1, let $p_\varepsilon(m)$ be a polynomial in m and an upper bound on the expected number of fitness evaluations for the (local) EA to find a $(1+\varepsilon)$-approximation. The (local) (1+1) EA with an efficient implementation of the mutation operator and the fitness function that halts after $4p_\varepsilon(m)$ fitness evaluations is a PRAS for the maximum matching problem, i. e., it finds a $(1+\varepsilon)$-optimal solution with probability at least* $3/4$.

3 Paths

Here, we prove that the (local) (1+1) EA finds maximum matchings for graphs consisting of a path of m edges in expected polynomial time. Among all graphs, these graphs allow the maximum length m for an augmenting path if m is odd. We prepare our analysis by describing the matchings on a fitness level distinct from the level of all maximum matchings. During the exploration of a fitness level, the number of disjoint augmenting paths is unchanged; otherwise the matching size would change, too. However, individual augmenting paths may vanish and new augmenting paths are created at the same time. Figure 1 depicts such a mutation. Solid lines indicate matching edges, dashed lines indicate free edges; the path's names after the mutation step are chosen arbitrarily. The shortest augmenting paths are edges with two exposed endpoints. We term these edges *selectable*, e. g., A' is a path consisting of a single selectable edge.

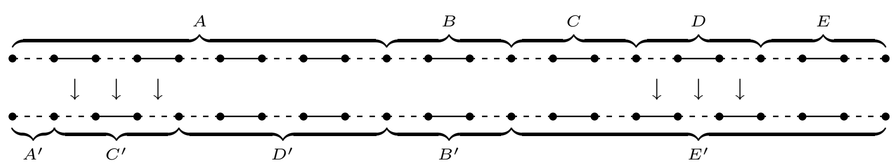

Fig. 1. Mutation step.

With Lemma 1, we can assume that the (local) (1+1) EA has arrived at a matching after expected time $O(m^2)$ and there are at most $\lceil m/2 \rceil$ fitness levels left to climb. At each point of time during the exploration of a fitness level, we focus an augmenting path P but consider only relevant steps. A *relevant* step alters P and produces a new string accepted by the EA. Furthermore, we distinguish two situations. In *Situation 1*, the current matching is not maximal, i. e., there exists some selectable edge e. The current matching can be improved by flipping exactly the right bit in the next step. We choose $P = \{e\}$ and for both EAs the probability that the next step is relevant (event R) is $\Theta(1/m)$. In *Situation 2*, the matching is maximal and, therefore, cannot be improved by a 1-bit flip. Shortest augmenting paths have length at least three. For all choices of P, the probability of a relevant step is $\Theta(1/m^2)$: It is lower bounded by the probability that only a specific pair of edges at one end of P flips and upper bounded by the probability that at least one of at most four edge pairs at both ends of P flip (only for the (1+1) EA there are some more possibilities where at least three edges in line flip). Clearly, both EAs have a not very small probability to leave the current fitness level in Situation 1, whereas for the (1+1) EA it is much harder to leave the level in Situation 2 and for the local (1+1) EA even impossible. The EAs enter a fitness level in either situation and may shift from one situation to the other several times until they finally leave the level. We name such a mutation step *improving*. As we have seen, at any time, the probability of a relevant step is at least $\Omega(1/m^2)$. Hence, the expected number of steps per relevant step is at most $O(m^2)$. If an expected number of T relevant steps is necessary to reach some target then the expected total number of steps is $\sum_{0\le t<\infty} \mathrm{E}(\#\text{steps} \mid T = t)\cdot \mathrm{Prob}(T = t) \le \sum_{0\le t<\infty} O(m^2)\cdot t \cdot \mathrm{Prob}(T = t) = O(m^2)\cdot \mathrm{E}(T)$. We use this property in the following way to show that it takes expected time $O(m^4)$ to find a maximum matching on a path with m edges. The size of the maximum matching equals $\lceil m/2\rceil$. If the current matching size is $\lceil m/2\rceil - i$, there exist i disjoint augmenting path; one of length at most $\ell := m/i$. If an expected number of $O(\ell^2)$ relevant steps are sufficient to improve the matching by one edge then $\sum_{1\le i\le\lceil m/2\rceil} O((m/i)^2) = O(m^2)$ relevant steps are sufficient for the optimum.

As a beginning, we consider the local (1+1) EA and demonstrate that central ideas of our proofs are easy to capture. Our analysis of the (1+1) EA only takes advantage of mutation steps where at most two bits flip, too. All other mutation steps only complicate the analysis and lengthen proofs considerably.

Theorem 2. *For a path of m edges, the expected runtime of the local (1+1) EA is $O(m^4)$.*

Proof. With our foregoing remarks we only have to show that the expected number of relevant steps to leave the current fitness level is $O(\ell^2)$. Consider Situation 1 and let A be the event that only e flips in the next step and thereby improves the matching, i. e., A implies R. Then $\mathrm{Prob}_R(A) := \mathrm{Prob}(A \mid R) = \mathrm{Prob}(A)/\mathrm{Prob}(R) = \Omega(1/m)/O(1/m) = \Omega(1)$ and the expected total number of relevant steps the EA spends in Situation 1 is $O(1)$. Let B be the event that the next step is not improving and leads to Situation 2; again B implies R. We want to bound $\mathrm{Prob}_R(B) := \mathrm{Prob}(B \mid R) = \mathrm{Prob}(B)/\mathrm{Prob}(R)$ from above. Only a mixed 2-bit flip can preserve the matching size. By definition, the selectable edge e has no neighbor in the matching. Hence, one of at most two neighbored pairs next to e has to flip. Thus $\mathrm{Prob}_R(B) = O(1/m^2)/\Omega(1/m) = O(1/m)$. As A and B are disjoint events, the conditional probability to improve the matching when leaving Situation 1 in a relevant step is $\mathrm{Prob}_R(A \mid A\cup B) = \mathrm{Prob}_R(A)/(\mathrm{Prob}_R(A) + \mathrm{Prob}_R(B)) = 1 - \Omega(1/m)$. Thus the expected number of times the EA leaves Situation 1 is at most $1 + O(1/m)$. Consequently, the expected number of times the EA leaves Situation 2 is bounded by $1+O(1/m) = O(1)$, too. Now it suffices to show that the expected number of relevant steps to leave Situation 2 is $O(\ell^2)$. To this end, the rest of this proof shows for some constants c and $\alpha > 0$, the probability to leave Situation 2 within $c\ell^2$ relevant steps is bounded below by α. Since the proof will be independent of the initial string when the EA enters Situation 2, it implies the $O(\ell^2)$ bound for leaving Situation 2. Having this, the expected number of relevant steps to leave the level is dominated by the product of the expected number of steps to leave Situation 2 and the number of times to leave Situation 2. Since in our analysis both numbers are upper bounded by independent random variables, expectations multiply to $O(\ell^2)$.

In Situation 2, there are no selectable edges. Straightforward considerations show that only pairs of neighbored edges located at one end of an alternating path can flip in an accepted step. Consider a phase of $c\ell^2$ relevant steps, possibly finished prematurely when an augmenting path of length one is created. Within the phase, augmenting paths have minimum length three. We focus attention to an augmenting path P whose initial length is at most ℓ and estimate the probability that P shrinks to length one and finishes the phase. If another augmenting path accomplishes this before P does, so much the better. We call a relevant step a *success* if it shortens P (by two edges). Since P can always shrink by two edges at both ends but sometimes cannot grow at both ends, the probability of a success is lower bounded by $1/2$ in each relevant step. As the initial length of P is at most ℓ, $\ell/4$ more successes than the expected value guarantee that P shrinks to length one in that time. We want to estimate the probability of at least $(1/2)c\ell^2 + (1/4)\ell$ successes within $c\ell^2$ relevant steps. With c chosen sufficiently large, $N := c\ell^2$ and b a constant, the probability of exactly k successful steps is

$$\binom{N}{k}\left(\tfrac{1}{2}\right)^k\left(1-\tfrac{1}{2}\right)^{N-k} \le \binom{N}{N/2}2^{-N} \le \frac{\sqrt{3\pi}\mathrm{e}^{-N}N^{N+1/2}2^{-N}}{\left(\sqrt{2\pi}\mathrm{e}^{-N/2}(N/2)^{N/2+1/2}\right)^2} = bN^{-1/2} \le \tfrac{1}{2\ell}.$$

The probability of less than $(1/2)c\ell^2 + (1/4)\ell$ successes is bounded above by Prob(less than $(1/2)c\ell^2$ successes) $+ \sum_{k=(1/2)c\ell^2}^{(1/2)c\ell^2+(1/4)\ell-1} \frac{1}{2\ell} \le \frac{1}{2} + \frac{\ell}{4}\cdot\frac{1}{2\ell} = \frac{5}{8}$. □

Theorem 3. *For a path of m edges, the (1+1) EA's expected runtime is $O(m^4)$.*

Proof. As in the previous proof, we only have to show that the expected number of relevant steps to leave a level is $O(\ell^2)$. In Situation 1, the probability that a relevant step is improving again is $\mathrm{Prob}_R(A) = \Omega(1)$. A necessary condition to move to Situation 2 in a relevant step is that e or at least one of at most two neighbors of e is turned into a matching edge. Thus $\mathrm{Prob}_R(B) = O(1/m^2)/\Omega(1/m) = O(1/m)$. As before, the expected total number of relevant steps in Situation 1 is $O(1)$ and the expected number of times the (1+1) EA leaves Situation 2 is at most $1 + O(1/m)$. It suffices to show that $c\ell^2$ relevant steps succeed in leaving Situation 2 with a probability $\alpha = \Omega(1)$.

In Situation 2, we ignore improving steps; they may take place and only shorten the time to leave the fitness level. Again we focus on an augmenting path P whose initial length is at most ℓ and consider a phase of $c\ell^2$ relevant steps. The phase is finished prematurely when P or another augmenting path shrinks to length one. The (1+1) EA allows mutation steps where the path's length $|P|$

changes by more than two edges or P vanishes completely as depicted in Fig. 1. The following properties ensure that $|P|$ never changes by more than two edges (implying none or two) in any step of the phase. Let x and y be the vertices at the current endpoints of P. Furthermore, let $E_x = \big\{\{u, v\} \in E \bigm| \mathrm{dist}(x, u) \leq 3\big\}$ be the set of edges where one endpoint has at most distance three to x, analogously for y (Fig. 2). The first property is that no step is accepted and flips more than

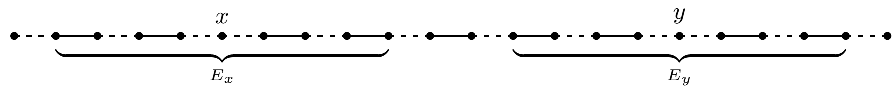

Fig. 2. Environments E_x and E_y.

three edges in $E_x \cup E_y$. The second property is that no step flips three or more edges in line on the extended path $P' := P \cup E_x \cup E_y$ and is accepted. We call steps that respect both properties or finish the phase *clean*. Obviously, we only have to ensure that all $c\ell^2$ relevant steps are clean and call this event C. We show that, given a step is relevant, it is clean with a probability $1 - O(1/m^2)$ and start by considering unconditioned probabilities. A necessary event to violate the first property is that a subset of four edges of $E_x \cup E_y$ flips and the corresponding probability is at most $\binom{16}{4}/m^4 = O(1/m^4)$ for a single step. For the second property, let k be the length of a longest block of flipping edges of P'. For $k \geq 5$, the probability that the block flips is at most $m \cdot 1/m^5 = O(1/m^4)$. Now let $k = 4$. Four flipping edges where x or y is an endpoint of the block is already excluded by the first property. All other blocks of length four violate the matching condition if they flip. A mutation step where $k = 3$ produces a local surplus of either one free edge or one matching edge in the block. The surplus must be balanced outside the block; otherwise the step is not accepted. To compensate a surplus of one free edge, another free edge must flip into a matching edge elsewhere. Since there are no selectable edges in Situation 2, in fact another block of at least three edges disjoint to the first block has to flip, too. This results in a probability of at most $(m \cdot 1/m^3)^2 = O(1/m^4)$. If a block of two free edges and a matching edge flips, either a non-matching is produced or the matching is improved locally in the block. If in the latter case the surplus of one matching edge is not balanced elsewhere, the phase is finished. Otherwise, either another single matching edge flips and thereby finishes the phase, too, or another block of at least three edges flips. The probability of the last event again is $O(1/m^4)$. Let D be the event of a clean step. Thus $\mathrm{Prob}(\overline{D}) = O(1/m^4)$ and given that a step is relevant the probability is $\mathrm{Prob}(\overline{D} \mid R) = \frac{\mathrm{Prob}(\overline{D} \cap R)}{\mathrm{Prob}(R)} \leq \frac{\mathrm{Prob}(\overline{D})}{\mathrm{Prob}(R)} = O(1/m^2)$. Hence, $\mathrm{Prob}(D \mid R) = 1 - O(1/m^2)$ and for a certain constant d and $m^2 \geq 2d$, $\mathrm{Prob}(C) \geq (1 - d/m^2)^{c\ell^2} \geq (1 - d/m^2)^{cm^2} \geq e^{-2cd} = \Omega(1)$ holds.

In the proof for the local (1+1) EA we have already seen, for initial path length at most ℓ, $c\ell^2$ relevant steps produce $\ell/4$ more successes than the expected value and succeed in decreasing the path length to one with probability at least 3/8 if c is sufficiently large and a relevant step shortens the path with probability at least 1/2. We call the event, given $c\ell^2$ relevant and clean steps, these steps succeed in decreasing the path to length one, event S. Then the success probability of a phase is at least $\mathrm{Prob}(C \cap S) = \mathrm{Prob}(C) \cdot \mathrm{Prob}(S \mid C)$. Given that a relevant step in Situation 2 is clean implies that it either finishes a phase or it flips one and only one pair of neighbored edges at one end of P (and perhaps some more edges not affecting P). In the latter case, the probability to flip a pair of edges shortening the path is at least $1/2$ and the probability to lengthen the path is at most $1/2$, since there are at least two shortening pairs and sometimes only one lengthening pair. Thus $\mathrm{Prob}(S \mid C) \geq 3/8$. □

We discuss the results of Theorem 2 and 3. Paths are difficult since augmenting paths tend to be rather long in the final stages of optimization. The (1+1) EA can cope with this difficulty. Paths are easy since there are not many possibilities to lengthen an augmenting path. The time bound $O(m^4) = O(n^4)$ is huge but can be explained by the characteristics of general (and somehow blind) search. If we consider a step relevant if it alters any augmenting path, there are many irrelevant steps, including steps which are rejected. In the case of $O(1)$ augmenting paths and no selectable edge a step is relevant only with a probability of $\Theta(1/m^2)$. The expected number of relevant steps is bounded by only $O(m^2) = O(n^2)$. Indeed, the search on the level of second-best matchings is already responsible for this. Since lengthenings and shortenings of the augmenting path have almost always the same probability for the local (1+1) EA, we are in a situation of fair coin tosses and have to wait for the first point of time where we have $\Theta(m)$ more heads than tails and this takes $\Theta(m^2)$ coin tosses with large probability. This implies that our bounds are tight if we have one augmenting path of length $\Theta(m)$. The situation is more difficult for the (1+1) EA. It is likely that from time to time several simultaneously flipping bits change the scenario drastically. We have no real control of these events. However, by focussing on one augmenting path we can ignore these events for the other paths and can prove a bound on the probability of a bad event for the selected augmenting path which is small enough that we may interpret the event as a bad phase. The expected number of phases until a phase is successful can be bounded by a constant. These arguments imply that we overestimate the expected time on the fitness levels with small matchings and many augmenting paths. This does not matter since the last improvement has an expected time of $\Theta(m^4)$ if we start with an augmenting path of length $\Theta(m)$.

4 Example with Exponential Time

After having seen that the (local) (1+1) EA computes maximum matchings on very simple graphs efficiently, we present a class of bipartite graphs where both EAs have an exponential expected optimization time. The graph $G_{h,\ell}$ is defined on $n = h\cdot(\ell+1)$ nodes where $\ell \geq 3$ is odd. To describe the graphs we consider the nodes as grid points (i, j), $1 \leq i \leq h$, $0 \leq j \leq \ell$. The nodes $(\cdot, j)$ belong to the jth column. Between column j, j even, and $j+1$ there are the edges $\{(i, j), (i, j+1)\}$ and between column j, j odd, and $j+1$ we have all edges of a complete bipartite graph. Fig. 3 depicts $G_{3,11}$ and its unique perfect matching. Since every perfect

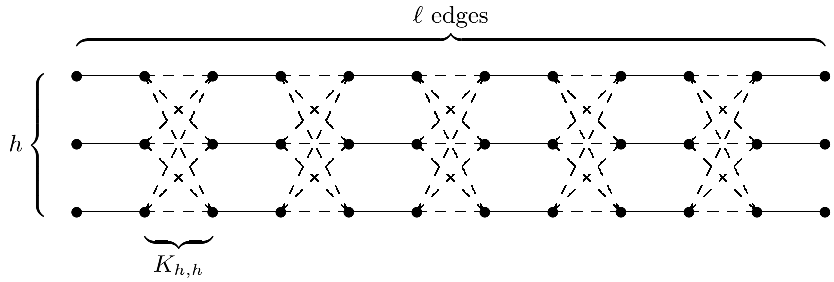

Fig. 3. The graph $G_{h,\ell}$ and its perfect matching.

matching must cover the nodes in column 0, all edges $\{(i,0),(i,1)\}$ belong to every perfect matching. Therefore, all edges $\{(\cdot,1),(\cdot,2)\}$ do not belong to any perfect matching and all edges $\{(i,2),(i,3)\}$ must be contained in every perfect matching. By induction on ℓ, it follows that every $G_{h,\ell}$ has a unique perfect matching consisting of all edges $\{(i,j),(i,j+1)\}$ where $1 \le i \le h$ and j is even. Sasaki and Hajek (1988) have proved that simulated annealing has an exponential expected optimization time on these graphs for $h = \ell$. Our result is the following one.

Theorem 4. *The local (1+1) EA has an exponential expected optimization time $2^{\Omega(\ell)}$ on $G_{h,\ell}$ if $h \ge 2$. For the (1+1) EA the expected optimization time is $2^{\Omega(\ell)}$ if $h \ge 3$ and $2^{\Omega(\ell^{\varepsilon})}$ for a certain $\varepsilon > 0$ if $h = 2$.*

It is interesting that our result holds also in the case of constant $h = 2$ where the degree of the graph is bounded by 3. Hence, the (local) (1+1) EA is not successful on graphs of constant degree. Observe that we obtain a path if $h = 1$.

We are mostly interested in the situation where the algorithm has found an almost optimal matching of size $n/2 - 1$. Then it is also easy to see that there exists exactly one augmenting path: Assume there are at least two augmenting paths P and Q. Using P or Q to improve the matching would result in two different maximum matchings, i. e., perfect matchings; a contradiction. So let P be the only augmenting path. Its length ℓ' is an odd number bounded by ℓ. P contains the nodes $(i_0,j),(i_1,j+1),\dots,(i_{\ell'},j+\ell')$, where $i_0 = i_1, i_2 = i_3, \dots, i_{\ell'-1} = i_{\ell'}$ and j is even, i. e., it runs from left to right possibly changing the level (see Fig. 4). To see this, just observe the following easy fact. Since $P = (e_1,\dots,e_{\ell'})$ is an augmenting path, its free edges $\{e_1,e_3,e_5,\dots,e_{\ell'}\}$ belong to the perfect matching. The endpoints of these edges are only linked to nodes in the next column. Thus one of these links is a selected edge of P (except for the first and last point). So the path P runs from left to right from (i_0,j) to $(i_{\ell'},j+\ell')$.

The main observation is that an accepted 2-bit flip can shorten or lengthen the augmenting path at either endpoint. However, at each endpoint (if not in column 0 or ℓ) there are h possibilities to lengthen the augmenting path and only one possibility to shorten it. Only if $h = 2$ and one endpoint is in column 0 or ℓ, there are 2 possibilities to lengthen the augmenting path and the same number of possibilities to shorten it. This explains why we sometimes have to consider the case $h = 2$ separately in the proof of Theorem 4. From a more global point of

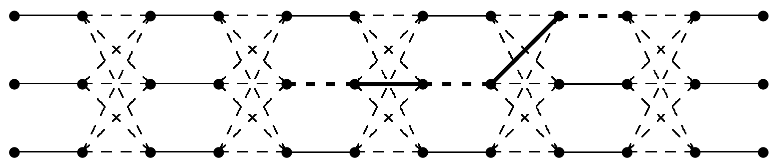

Fig. 4. An almost perfect matching and its augmenting path.

view, we may consider "semi-augmenting" paths, i. e., alternating paths starting at a free node which cannot be lengthened to an augmenting path. The number of semi-augmenting paths is exponential (if ℓ' is not close to ℓ). The (local) (1+1) EA searches more or less locally and cannot distinguish immediately between semi-augmenting paths and augmenting paths. Our conjecture is that the presence of exponentially many semi-augmenting paths and only polynomially many augmenting paths at many points of time prevents the (local) (1+1) EA from being efficient. This also explains why paths are easy and why trees should be easy for the (local) (1+1) EA. The proof of Theorem 4 follows this intuition: The first matching found is typically not the perfect matching. The conjecture is that the probability is even close to 1, at least $1 - o(1)$. However, Lemma 2 is strong enough to prove Theorem 4. Then Lemma 3 guarantees that the algorithms find almost perfect matchings of size $n/2 - 1$ before the optimum is found with a probability close to 1. In this situation, the vast number of semi-augmenting paths hinder the algorithms from being successful within reasonable expected time. By Lemma 4, the probability that the augmenting path extends to its maximum length during the exploration is not too small. Finally, Lemma 5 shows that it takes exponential time to improve the matching in this situation.

Lemma 2. *With a probability of $\Omega(1/h)$ the first matching found by the (local) (1+1) EA is not perfect.*

Lemma 2 follows from Claim 1 and Claim 2.

Claim 1. The first search point is with overwhelming probability a non-matching.

Proof. The first search point is chosen randomly. Hence, each inner point with degree $h+1$ has a probability of $1-(1/2)^{h+1}(h+2) \geq 1/2$ to be touched by at least two chosen edges. Inner nodes in the same column or with column distance at least 2 are independent. Hence, the probability to start with a matching is exponentially small. □

Claim 2. Starting with a non-matching, with a probability of $\Omega(1/h)$ the first matching is not perfect.

Proof. We prove the lemma separately for the local (1+1) EA and the (1+1) EA.

Local (1+1) EA. We view the configuration graph of the process as a tree. The root is the initial configuration and the nodes on level d represent all configurations reachable in exactly d steps of the algorithm. Edges lead only from Level d to Level $d+1$ and are marked with transition probabilities. All leaves of this infinite tree represent configurations where a matching is reached for the first time and all inner nodes represent configurations corresponding to non-matchings. By $\mathrm{Prob}(v)$ we denote the probability to reach a specific node v. Then $\mathrm{Prob}(v)$ is the product of all probabilities assigned to edges on the path from the root to v. For short, we name leaves representing the perfect matching *p-leaves* and all other leaves representing non-perfect matchings *n-leaves*. We want to show that

$$\sum_{v \text{ is a p-leaf}} \mathrm{Prob}(v) \;=\; O(h) \cdot \sum_{v \text{ is an n-leaf}} \mathrm{Prob}(v).$$

This implies the claim. Both sums in the last equation converge absolutely; we may split and reorder the terms on the left-hand side to compare the sums. We do this by assigning the probability of every p-leaf to one or more n-leaves in its neighborhood and ensure that the sum of probabilities assigned to an n-leaf v is $O(h) \cdot \mathrm{Prob}(v)$. For the following case inspection we first observe that every inner node of the tree has at most one p-leaf as descendant. The reason is that the perfect matching is unique and, therefore, the required mutation step is determined uniquely.

Let b be a p-leaf and a its predecessor in the configuration tree. For convenience, a and b denote nodes of the tree and also the corresponding sets of selected edges of $G_{h,\ell}$, i.e., b also denotes the perfect matching. If $H(a,b)=1$ then a is a superset of b since a is a non-matching. Hence, a has one edge e more than b and a has $n/2$ further descendants $b'_1, \ldots, b'_{n/2}$ where e and another edge of a flip. Obviously, all b'_i are n-leaves and $\mathrm{Prob}(b'_i) = \mathrm{Prob}(a)/m \cdot 1/(m-1)$. The probability to reach b equals $\mathrm{Prob}(a)/(2m)$. We assign $\mathrm{Prob}(b)$ uniformly to all b'_i such that each b'_i gets a portion of $\mathrm{Prob}(a)/m \cdot 1/n$. Thus the probability assigned to each b'_i is only by a factor $(m-1)/n = \Theta(\ell h^2)/\Theta(\ell h) = O(h)$ larger than $\mathrm{Prob}(b'_i)$.

Now let $H(a,b)=2$. Because a is not a matching, a is not a subset of b. This implies that the edge (a,b) of the configuration tree represents a mutation where at least one selected edge flips. The first subcase is that (a,b) denotes a mutation where a free edge e and a selected edge e' flip. Then b has a sibling b' where only e' flips and b' is an n-leaf. As $\mathrm{Prob}(b) = \mathrm{Prob}(a)/(m(m-1))$ and $\mathrm{Prob}(b') = \mathrm{Prob}(a)/(2m)$ we can assign $\mathrm{Prob}(b)$ to b' and there is even a large capacity left at b'. The second subcase is where (a,b) is a mutation step with two selected edges e and e' flipping. Clearly, a is a superset of b and there is no other sibling of b that is a leaf. But there exist $n/2$ siblings $b'_1, \ldots, b'_{n/2}$ that correspond to the first subcase and are derived from a by flipping e and any of $n/2$ other selected edges $e'' \neq e'$. Each b'_i has a descendant c_i obtained by flipping e' and e'', a p-leaf, and another descendant c'_i derived by flipping only e', an n-leaf. That means, the edge (b'_i, c_i) is the same situation as the first subcase implying that c'_i already obtains $\mathrm{Prob}(c_i) = \mathrm{Prob}(a)/(m^2(m-1)^2)$. Now we assign $\mathrm{Prob}(b)$

uniformly to all c_i', i.e., each c_i' additionally gets $(2/n) \cdot \mathrm{Prob}(a)/(m(m-1))$. This can happen at most once, since b_i' has at most one sibling b that is a p-leaf and assigns probability to a descendant of b_i'. Altogether, the probability assigned to c_i' is at most

$$\frac{\mathrm{Prob}(a)}{m^2(m-1)^2}+\frac{2\mathrm{Prob}(a)}{nm(m-1)}=\frac{\mathrm{Prob}(a)}{2m^2(m-1)}\cdot\Big(\frac{2}{m-1}+\frac{4m}{n}\Big)=\frac{\mathrm{Prob}(a)}{2m^2(m-1)}\cdot\Theta(h)$$

and at most by a factor $O(h)$ larger than $\mathrm{Prob}(c_i') = \mathrm{Prob}(a)/(2m^2(m-1))$.

(1+1) EA. We consider a phase of cm^2 steps, c a constant large enough. Then, with a probability exponentially close to 1, we have among these steps at least $(c/4)m^2$ steps flipping one bit each. By the proof of Lemma 1 this implies a probability of at least 1/2 of finding a matching. Moreover, with a probability exponentially close to 1, among these steps there is no step flipping at least $\ell^{1/2}$ 0-bits into 1-bits. (Since the probability to flip *any* $\ell^{1/2}$ bits in a step is exponentially small.) Now we investigate the step creating the first matching. If the matching is perfect, the previous search point contains already at least $(\ell+1)h/2 - \ell^{1/2} + 1$ of the edges of the perfect matching. The probability of a step flipping additionally one of the edges of the perfect matching is at most by a factor of $\Omega(\ell h/m) = \Omega(1/h)$ smaller than the probability of the step expected $(m = \Theta(\ell h^2))$. Hence, with a probability of at least $\Omega(1/h)$ the first matching is not perfect. □

Lemma 3. *Assuming the (local) (1+1) EA finds first a non-perfect matching, the probability to find an almost perfect matching, i.e., a matching of size $n/2-1$, before finding the perfect matching a^* is $1 - O(1/m)$.*

Proof. Assume the (1+1) EA's current matching a is not perfect and let a^* denote the perfect matching. If $H(a, a^*) = 1$ then a is almost perfect.

Local (1+1) EA. If $H(a, a^*) > 2$ the probability to find the optimum in the next step is 0, whereas the probability to find some almost perfect matching may be positive. If $H(a, a^*) = 2$ then clearly a is not a matching of size $n/2$ since a^* is the only perfect matching. We may consider the case where a is a matching of size $n/2 - 1$. Then a is almost perfect and there is nothing to prove. However, almost perfect matchings with a Hamming distance of 2 to the optimum do not exist since augmenting paths have odd length. Thus, we conclude a is a matching of size $n/2-2$ and, therefore, a subset of a^*. The probability to create a^* from a in the next step is $1/(m(m-1))$ and the probability to create an almost perfect matching is $1/m$. In any situation, the probability to create the perfect matching in the next step is at least by a factor $1/(m-1)$ smaller, i.e., the probability to create an almost perfect matching first is at least $1 - 1/m$.

(1+1) EA. If $H(a, a^*) =: d \geq 2$, the probability of creating a^* in the next step equals $(1/m)^d(1-1/m)^{m-d}$. As a is a matching and $d \geq 2$, there are at least two edges of a^* which do not belong to a. If exactly one of them does

not flip and everything else works as in the mutation $a \to a^*$, we get an almost perfect matching. The probability of such an event is $2(1/m)^{d-1}(1-1/m)^{m-d} \geq 2m(1/m)^d(1-1/m)^{m-d} \geq m(1/m)^d(1-1/m)m-d$. Therefore, the probability of creating a^* before creating any second-best matching is at most $1/(m+1) = O(1/m)$. □

Lemma 4. *Starting with an almost perfect matching, the probability that the augmenting path extends to its maximal length ℓ before the matching is improved is $\Omega(h/m)$.*

Assume that the (local) (1+1) EA has found a matching of size $n/2-1$. Let ℓ_1 be the length of the unique augmenting path with respect to the matching. We conjecture that ℓ_1 is with probability $1-o(1)$ large, i.e., at least ℓ^ε for some $\varepsilon > 0$. As a substitute of this unproven conjecture we use the following arguments. Before constructing the perfect matching the (1+1) EA searches on the plateau of matchings of size $n/2-1$, since no other search point is accepted. In order to investigate the search on this plateau, we describe the random search point in step k by X_k and its value by x_k. Moreover, let L_k be the random length of the augmenting path of X_k and ℓ_k the corresponding value of L_k. The search stops iff $\ell_k = 0$. Let T be this stopping time. Before that, ℓ_k is an odd number bounded above by ℓ. Claim 3 and Claim 4 imply Lemma 4.

Claim 3. If $\ell_1 = 1$, the probability that $\ell_k \geq 3$ for some $k < T$ is $\Omega(h/m)$.

Proof. Obviously, both algorithms accept almost perfect and perfect matchings only. As long as $\ell_k = 1$, there is a selectable edge e^* and the probability of creating the perfect matching equals $1/(2m)$ and $(1/m)(1-1/m)^{m-1}$ for the local (1+1) EA and the (1+1) EA, respectively. However, there are at least h pairs $\{e', e''\}$ such that e' is free, e'' is chosen, and (e^*, e', e'') is a path. If exactly e' and e'' flip, we obtain an augmenting path of length 3. The probability that this happens for one of the h pairs equals $h/(m(m-1)) = 2h/(m-1) \cdot 1/(2m)$ for the local (1+1) EA and $h(1/m)^2(1-1/m)^{m-2} \geq (h/m)(1/m)(1-1/m)^{m-1}$ for the (1+1) EA. Therefore, the probability that $\ell_k = 3$ before $\ell_{k'} = 0$ for $k' < k$ is at least $2h/(2h+m-1) \geq 2h/(3m) = \Omega(h/m)$ for the local (1+1) EA. For the (1+1) EA, this probability is at least $h/(m+h) \geq h/(2m) = \Omega(h/m)$. □

Claim 4. If $\ell_1 \geq 3$, the probability that $\ell_k = \ell$ for some $k < T$ is $\Omega(1)$.

In the proof of this claim, we refer to the *ruin problem.* Alice owns A \$ and Bob B \$. They play a coin-tossing game with a probability of $p \neq 1/2$ that Alice wins a round in this game, i.e., Bob pays a dollar to Alice. Let $t := (1-p)/p$. Then Alice wins, i.e., she has $(A+B)$ \$ before being ruined, with a probability of $(1-t^A)/(1-t^{A+B}) = 1 - t^A(1-t^A)/(1-t^{A+B})$ (e.g., Feller (1971)).

Proof (Claim 4). Again we prove the claim for both algorithms separately. Additionally, we distinguish the cases $h \geq 3$ and $h = 2$. However, the case $h = 2$ can be viewed as a worst-case when proving the stated lower bound and one can skip the case $h \geq 3$. Note that if $\ell_k \geq 3$, only steps where the same number of zeros and ones flip can be accepted, except for the special case for the (1+1) EA where all edges of the augmenting path flip.

Local (1+1) EA, $h \geq 3$. Let $b_k = (\ell_k - 1)/2$. The local (1+1) EA can change the b-value by at most 1 as long as its value is positive. Obviously, $b_1 \geq 1$ and pessimistically $b_1 = 1$. We are interested in the probability of reaching the maximal b-value $\ell/2$ before the value 0. There are two 2-bit flips decreasing the b-value by 1 and there are at least h (one endpoint of the augmenting path can be in column 0 or ℓ) 2-bit flips increasing the b-value. Hence, the next step changing the b-value leads to $b-1$ with a probability of at most $2/(h+2) \leq 2/5$ and leads to $b+1$ with a probability of at least $h/(h+2) \geq 3/5$. We have an unfair game and can apply the result of the gambler's ruin problem. With $A = 1$ and $B = (\ell-1)/2-1$ the probability that the b-value is $(\ell-1)/2$ before it drops to 0 is $(1-t)/(1-t^{(\ell-1)/2})$. Since $t = 2/h \leq 2/3$, this probability is at least $1/3$ and for general h it is at least $1 - O(1/h)$. Moreover, if ℓ_1 is not too small, this probability is even close to 1.

Local (1+1) EA, $h = 2$. W.l.o.g. let $\ell \equiv 1 \bmod 4$. In the beginning, there are at least two 2-bit flips increasing ℓ_k and exactly two 2-bit flips decreasing ℓ_k, i.e., the first accepted mutation increases ℓ_k with a probability of at least $1/2$. Thus with a probability of at least $1/2$, we reach an ℓ_k-value of at least 5 before an ℓ_k-value of 1. Pessimistically assuming we have reached the ℓ_k-value exactly 5, we consider only *relevant* steps, i.e., steps changing the length of the path, and group these steps in pairs of successive relevant steps. That means, the first and second, the third and fourth relevant step and so on are a pair. Observe that a pair can cause a net change of the path length of either -4, 0, or 4. Let $b_k = (\ell_k - 1)/4$, for $\ell_k \equiv 1 \bmod 4$. Each relevant step is increasing with a probability of at least $1/2$ and, therefore, a pair increases the b-value with a probability of at least $1/4$. We have to be more careful for an upper bound on the probability of a pair decreasing the b-value which should be less than $1/4$. We pessimistically assume that the path is in marginal position in the first step, i.e., one endpoint is in column 0 or ℓ. Now only four different 2-bit mutations can be accepted in the first step. With a probability of $1/4$ the first relevant step decreases the length *and* preserves the marginal position and with a probability of $1/4$ the first relevant step decreases the length such that thereafter the path is not in marginal position. In the first case, the next step is decreasing with a probability of $1/2$ and in the second case the next relevant step is decreasing with a probability of $1/3$. Thus $1/4 \cdot 1/2 + 1/4 \cdot 1/3 = 5/24$ is an upper bound on the probability of a decreasing pair. We consider only relevant pairs, i.e., pairs changing the the b-value by -1 or 1. The conditional probability of a decreasing pair, given the pair is relevant, is at most $5/11$ and for an increasing pair it is $6/11$. Again we have a coin-tossing game with an unfair coin with $t = 5/6$, $A = 1$, and $B = (\ell-1)/4-1$. The probability to reach a b-value $(\ell-1)/4$, before 0 is $(1-5/6)/(1-(5/6)^{(\ell-1)/4} \geq 1/6$. Altogether, the success probability is at least $1/12$.

(1+1) EA, $h \geq 3$. We investigate some relevant probabilities. As long as $\ell_k < \ell$ we have at least h 2-bit flips increasing ℓ_k by 2 and the corresponding probability is at least $h(1/m)^2(1-1/m)^{m-2}$. The probability to create the perfect matching

equals $(1/m)^{\ell_k}(1-1/m)^{m-\ell_k} \leq (1/m)^{\ell_k}$. The probability that $\ell_{k+1} - \ell_k \leq -4$ is bounded above by $3/m^4$, since (at least) either the first four edges of the augmenting path P_k have to flip or the last four edges of P_k or the first two and the last two edges of P_k. Finally, we carefully estimate the probability that $\ell_{k+1} - \ell_k = -2$. This event happens only if exactly the first two edges of P_k or exactly the last two edges of P_k flip or if the first four, the last four, or the first two and the last two edges of P_k (and some more edges) flip. For the last event we pessimistically assume that $\ell_{k+1} - \ell_k \leq -4$ and this event has been considered above. Hence, under this assumption the probability of $\ell_{k+1} - \ell_k = -2$ equals $2 \cdot (1/m)^2(1-1/m)^{m-2}$. The event of creating the perfect matching has a "large" probability only if $\ell_k \leq 3$. Hence, we first consider the probability of reaching an ℓ_k-value of at least 5 before reaching a value of at most 1. From the results above this probability is bounded below by $1/2 - o(1)$.

In a step where $\ell_{k+1} - \ell_k \geq 2$ we pessimistically assume that $\ell_{k+1} - \ell_k = 2$. Thereby, we only decrease the probability to reach an ℓ_k-value of ℓ first. For a moment, we additionally assume that for each step also $\ell_{k+1} - \ell_k \geq -2$, i. e., we ignore all other steps. Then a step is called *relevant* if it changes the length of the augmenting path. The next relevant step increases and decreases the ℓ_k-value by 2 with a probability of at least $h/(h+2) \geq 3/5$ and at most $2/(h+2) \leq 2/5$, respectively. With the arguments of the ruin problem, we know that under this assumption the probability of obtaining an augmenting path of length ℓ before obtaining an augmenting path of length at most 3 is at least $(1/3)$.

In order to drop the last assumption, we investigate a phase of $m^{7/2}$ steps of the algorithm. We state that in the phase there is no step such that $\ell_{k+1} - \ell_k \leq -4$ with a probability close to 1 and the coin-tossing game is finished within the phase with a probability close to 1. Let A be the event of winning the coin-tossing game and let B be the event that the phase contains no step where the ℓ_k-value decreases by at least 4. Then $\text{Prob}(B) = 1 - O(m^{-1/2})$. The probability of a relevant 2-bit flip is $\Omega(1/m^2)$. Let C be the event that we have $\Omega(m^{5/4})$ relevant 2-bit flips and the difference between increasing and decreasing 2-bit flips is $\Omega(m^{5/4})$. With a probability exponentially close to 1 we have $\Omega(m^{5/4})$ relevant 2-bit flips. The conditional probability of a relevant step to increase the length of the augmenting path is at least $3/5$. Hence, the probability that there is a surplus of $\Omega(m^{5/4}) = \Omega(\ell^{5/4})$ increasing 2-bit flips is exponentially close to 1. With already $\ell/2$ more increasing than decreasing 2-bit flips we have reached an ℓ_k-value of ℓ such that the coin-tossing game is surely finished. Thus, $\text{Prob}(C)$ is exponentially close to 1 and $\text{Prob}(B \cap C) = 1 - O(m^{-1/2})$. Now, $\text{Prob}(A \cap B \cap C) = \text{Prob}(A \mid B \cap C) \cdot \text{Prob}(B \cap C)$. Under the assumption $B \cap C$, the probability to reach an ℓ_k-value of ℓ before an ℓ_k-value of at most 3 is only increased. Moreover, reaching an ℓ_k-value of ℓ before an ℓ_k-value of at most 3 ensures that 3-bit flips cannot occur. Hence, $\text{Prob}(A \cap B \cap C) = 1/3 - O(m^{-1/2})$. Altogether, the probability of interest is bounded below by $1/6 - o(1)$.

(1+1) EA, $h = 2$. W.l.o.g. let $\ell \equiv 1 \bmod 4$. There are at least two 2-bit flips increasing ℓ_k by 2 and $2(1/m)^2(1-1/m)^{m-2}$ is a lower bound for the corresponding probability. If the augmenting path is not in marginal position then there are

even four 2-bit flips increasing ℓ_k and the probability is $4(1/m)^2(1-1/m)^{m-2}$. The probability that $|\ell_{k+1}-\ell_k| \geq 4$ is $O(1/m^4)$. The reason is that at least 4 edges in the following set E_k of edges have to flip. If the augmenting path starts in column i and ends in column j, the set contains all edges between column $i-4$ and $i+4$ and between column $j-4$ and $j+4$. Since $h=2$, $|E_k|$ is bounded by a constant and so is the number of ways to choose four edges from $|E_k|$. Again we assume that $\ell_{k+1}-\ell_k=-2$ only if either exactly the first two edges of P_k or exactly the last two edges of P_k flip. Otherwise, at least four edges of E_k have to flip and we pessimistically assume that $\ell_{k+1}-\ell_k<-2$. With this assumption, the probability that $\ell_{k+1}-\ell_k=-2$ is $2(1/m)^2(1-1/m)^{m-2}$. The probability of obtaining an augmenting path of length at least 9 before obtaining a path of length at most 1 is $1/8-o(1)$.

We investigate a phase of $m^{7/2}$ steps. Let B be the event that the phase has no step such that $|\ell_{k+1}-\ell_k| \geq 4$. Then $\text{Prob}(B)$ is $1-O(m^{-1/2})$. The probability of a *relevant step*, a 2-bit flip changing ℓ_k, is $\Omega(1/m^2)$ and with a probability exponentially close to 1 the phase contains $\Omega(m^{5/4})$ relevant steps. With the condition B, the probability of that many relevant steps only increases. Now we group the relevant steps in pairs, i.e., a pair consists of two consecutive relevant steps. We consider only *relevant pairs* consisting of either two increasing or two decreasing relevant steps. Relevant pairs change ℓ_k by either 4 or -4. The probability that a step in a pair is increasing is always at least $1/2$ and a pair is increasing with a probability of at least $1/4$. Thus with a probability exponentially close to 1 there are $\Omega(m^{5/4})$ relevant pairs. For an upper bound on the probability of a decreasing pair we pessimistically assume that the path is in marginal position in the first step, i.e., one endpoint is in column 0 or ℓ. With a probability of $1/4$ the first step decreases the length *and* preserves the marginal position. With a probability of $1/4$ the first step decreases the length such that thereafter the path is not in marginal position. In the first case, the next step is decreasing with a probability of $1/2$ and in the second case the next step is decreasing with a probability of $1/3$. Hence, $1/4\cdot 1/2+1/4\cdot 1/3=5/24$ is an upper bound on the probability of a decreasing pair. Now we pessimistically assume that the probability of an increasing pair is $1/4$ and the probability of a decreasing pair is $5/24$. With this assumption and given that a pair is relevant, the pair is increasing with a probability of at least $6/11$ and the probability that there are $\Omega(m^{5/4})$ more increasing pairs than decreasing pairs is exponentially close to 1. Let $\text{Prob}(C \mid B)$ be the probability that, given B, there are $\Omega(m^{5/4})$ relevant 2-bit flips forming $\Omega(m^{5/4})$ relevant pairs with a surplus of $\Omega(m^{5/4})=\Omega(\ell^{5/4})$ increasing pairs. Then $\text{Prob}(C \mid B)$ is exponentially close to 1 and $\text{Prob}(B\cap C)=\text{Prob}(C \mid B)\cdot\text{Prob}(B)=1-O(m^{-1/2})$. Again we consider a coin-tossing game and define the b-values in the following way. Let $b_k:=(\ell_k-5)/4$. The initial b-value is at least 1 and corresponds to an augmenting path of length 9. Each relevant pair changes the b-value by 1 or -1. The maximal b-value is $(\ell-5)/4$ and corresponds to a path of length ℓ. Then the probability $\text{Prob}(A)$ to reach a b-value of 0 before the maximal b-value in the coin-tossing game is $(1-5/6)/(1-5/6)^{(\ell-3)/4} \geq 1/6$. Now,

$\text{Prob}(A \cap B \cap C) = \text{Prob}(A \mid B \cap C) \cdot \text{Prob}(B \cap C)$. Condition $B \cap C$ ensures that the coin-tossing game is finished within the phase and only increases the probability to win the game. Thus, $\text{Prob}(A \cap B \cap C) = 1/6 - O(m^{-1/2})$. Altogether, the probability of interest is bounded below by $1/48 - o(1)$. □

With Lemma 2, 3, and 4, both algorithms reach an almost perfect matching where the path length ℓ_k is ℓ with a probability of $\Omega(1/m)$. For each fixed h we have $m = \Theta(\ell)$. In order to prove Theorem 4 it now suffices to prove the following lemma.

Lemma 5. *Starting with an almost perfect matching with an augmenting path of length ℓ, the probability that the local (1+1) EA finds the perfect matching within $2^{c\ell}$ steps, $c > 0$ an appropriate constant, is bounded by $2^{-\Omega(\ell)}$ if $h \geq 2$. In particular, the expected optimization time is $2^{\Omega(\ell)}$. The same holds for the (1+1) EA if $h \geq 3$. For $h = 2$, the (1+1) EA's expected optimization time is $2^{\Omega(\ell^{\varepsilon})}$ for a certain constant $\varepsilon > 0$.*

For the local (1+1) EA and both cases $h \geq 3$ and $h = 2$, the presented proofs are easy and almost identical. The proof for the (1+1) EA and $h \geq 3$ is much more involved but the arguments used do not obviously carry over to the case $h = 2$. We present a more direct approach for the (1+1) EA and $h = 2$. However, the exponential lower bound is weaker in this case.

Proof (Lemma 5).

Local (1+1) EA, $h \geq 3$. W.l.o.g. let $\ell \equiv 1 \bmod 4$. We reuse the b-values defined in the proof of Lemma 4 and start with the maximal b-value $(\ell-1)/2$. In order to reach the value 0 the value $(\ell-1)/4$ must be reached first. The probability to reach then $(\ell-1)/2$ before 0 is at least $1 - t^{(\ell-1)/4}(1 - t^{(\ell-1)/4})/(1 - t^{(\ell-1)/2})$. Since $t = 2/h \leq 2/3$, this probability is $1 - (2/h)^{\Theta(\ell)}$ and the probability of reaching 0 before $(\ell-1)/2$ can be bounded by $2^{-2c\ell}$, for a certain constant $c > 0$. We estimate the number of the first such phase that reaches the b_k-value 0 by a random variable T following the geometric distribution with parameter $p = 2^{-2c\ell}$. Then $\text{E}(T) = 2^{2c\ell}$ is a lower bound for the expected optimization time and the probability that T is at least $2^{c\ell}$ is bounded by

$$\left(1 - \frac{1}{2^{2c\ell}}\right)^{2^{c\ell}} \geq (\mathrm{e}^{-1})^{\frac{2^{cl}}{2^{2c\ell}-1}} = \mathrm{e}^{-2^{-\Omega(\ell)}} \geq 1 - 2^{-\Omega(\ell)}.$$

Local (1+1) EA, $h = 2$. For $h = 2$, we have defined the b-values differently. W.l.o.g. let $\ell \equiv 1 \bmod 8$. The maximal b-value is $(\ell-1)/4$. In order to reach 0, the local (1+1) EA first reaches the b-value $(\ell-1)/8$. The probability to reach $(\ell-1)/4$ again before reaching 0 is at least $1-t^{(\ell-1)/8}(1-t^{(\ell-1)/8})/(1-t^{(\ell-1)/4}) = 1 - (5/6)^{\Omega(\ell)}$. The probability to reach 0 first is $(5/6)^{\Omega(\ell)} = 2^{-\Omega(\ell)}$ and the expected number of phases is $2^{\Omega(\ell)}$. Analogously to the case $h \geq 2$, the expected optimization time is $2^{\Omega(\ell)}$ with an overwhelming probability.

(1+1) EA, $h \geq 3$. For the proof we apply methods due to Hajek (1982) which have been worked out by He and Yao (2001). Analyzing their proof it follows immediately that they have even proved a stronger result than stated, namely a result on the success probability and not only the expected waiting time for a success. We state this result in Theorem 5.

Theorem 5. *Let $X_0, X_1, X_2, \ldots$ be the random variables describing a Markov process and let $g\colon \mathbb{R} \to \mathbb{R}_0^+$, $0 \leq a(\ell) < b(\ell)$, $\lambda > 0$, $D \in \mathbb{R}$, and $p(\ell)$ a polynomial. Moreover, assume that*

$$g(X_0) \leq a(\ell) \text{ with probability 1,}$$

$$b(\ell) - a(\ell) = \Omega(\ell),$$

$$\mathrm{E}\big(\mathrm{e}^{\lambda(g(X_{t+1})-g(X_t))} \bigm| X_t = x \text{ and } a(\ell) < g(x) \leq b(\ell)\big) \leq 1 - 1/p(\ell), \text{ and}$$

$$\mathrm{E}\big(\mathrm{e}^{\lambda(g(X_{t+1})-a(\ell))} \bigm| X_t = x \text{ and } g(x) \leq a(\ell)\big) \leq D.$$

Let T be the smallest t where $g(X_t) \geq b(\ell)$. The probability that $T \leq B$ is bounded above by $D \cdot B \cdot \mathrm{e}^{\lambda(a(\ell)-b(\ell))} \cdot p(\ell)$.

Since $\lambda(a(\ell) - b(\ell)) = -\Omega(\ell)$ and p is a polynomial, this bound is exponentially small for $B = 2^{c\ell}$, $c \geq 0$ an appropriate constant.

Our initial Markov process is the (1+1) EA on our matching problem starting with an almost perfect matching with an augmenting path of length ℓ. In order to meet the conditions of Theorem 5 we define $g(X_k) := \ell - X_k$. Then $g(X_0) = 0$. Let $a(\ell) = 0$ and $b(\ell) = \ell - 3$. Then the first two conditions are fulfilled.

In order to simplify the calculations we replace the (1+1) EA by a Markov process on $\{0, 2, \ldots, \ell-3, \ell-1, \ell\}$. We do this by estimating probabilities to increase the g-value by larger values and probabilities to decrease the g-value by smaller values. The new Markov process will be time-homogeneous. In state 0, it is impossible to decrease the state. Otherwise, we only allow to go from state $2i$ to state $2(i-1)$ and ignore other decreasing steps. Since there are always at least h possibilities to lengthen the augmenting path, we estimate this probability below by

$$p_{-2} := h \cdot (1/m)^2 (1 - 1/m)^{m-2}.$$

There is the special case of reaching ℓ from state $2j$. Then exactly the edges of the augmenting path have to flip. This probability can be estimated above by

$$p^* := (1/m)^{\ell - 2j}.$$

Finally, we need an upper bound p_{2j} on the probability of increasing the state by $2j$ in one step. It is necessary to flip the $2i$ leftmost edges and the $2(j-i)$ rightmost edges of the augmenting path for some $i \in \{0, \ldots, j\}$. Hence,

$$p_{2j} := (j+1)(1/m)^{2j}$$

is a correct bound. For the special case $j = 1$ we need a better bound which is essentially smaller than p_{-2}. It is sufficient to argue as follows. There are exactly two possibilities flipping exactly two edges and otherwise we have to flip at least the $2i$, $0 \leq i \leq 2$, leftmost edges and the $4-2i$ rightmost edges of the augmenting path. Hence, we work with the new value

$$p_2 := 2 \cdot (1/m)^2 (1-1/m)^{m-2} + 3 \cdot (1/m)^4.$$

The remaining probability is

$$p_0 = 1 - (h+2)(1/m)^2(1-1/m)^{m-2} - O(m^{-3}).$$

Here we have used that due to our choice of $b(\ell) = \ell - 3 \geq 2j$ the bound for p^* is $O(m^{-3})$ in the situations described by the last two conditions of Theorem 5. Now we omit all steps not changing the state. Then the new transition probabilities are given by

$$q_{-2} = p_{-2}/(1-p_0), \quad q^* = p^*/(1-p_0), \quad \text{and } p_{2j} = p_{2j}/(1-p_0).$$

Now we have to estimate the following sum.

$$\mathrm{e}^{-2\lambda} q_{-2} + \mathrm{e}^{2\lambda} q_2 + \sum_{j \geq 2} \mathrm{e}^{2j\lambda} q_{2j} + \mathrm{e}^{(\ell-2i)\lambda} m^{-(\ell-2i)} \;/\; (1-p_0) \tag{1}$$

Note that $\lim_{\lambda \to 0} \mathrm{e}^{-2\lambda} q_{-2} + \mathrm{e}^{2\lambda} q_2 = q_{-2} + q_2 < 1$. First we show that there exist constants $\delta' > 0$ and $\lambda > 0$, such that $\mathrm{e}^{-2\lambda} q_{-2} + \mathrm{e}^{2\lambda} q_2 \leq 1 - \delta'$ for m large enough. We know that

$$p_{-2} - p_2 \;=\; (h-2)(1/m^2)(1-1/m)^{m-2} - 3(1/m^4) \;\geq\; \alpha(h-2)m^{-2}$$

for some $\alpha > 0$ and m large enough. Hence, $q_{-2} - q_2 \geq \beta$ for some $\beta > 0$ and m large enough. As λ approaches 0, $\mathrm{e}^{-2\lambda} \leq 1 - 2\lambda + c' \cdot \lambda^2$ and $\mathrm{e}^{2\lambda} \leq 1 + 2\lambda + c'' \cdot \lambda^2$ for certain constants c' and c''. This implies for $c = \max\{c', c''\}$ that

$$\begin{aligned}
\mathrm{e}^{-2\lambda} q_{-2} + \mathrm{e}^{2\lambda} q_2 \;&\leq\; (1 - 2\lambda + c \cdot \lambda^2) q_{-2} + (1 + 2\lambda + c \cdot \lambda^2) q_2 \\
&\leq\; q_{-2} + q_2 - 2\lambda \cdot (q_{-2} - q_2) + c \cdot \lambda^2 \cdot (q_{-2} + q_2) \\
&\leq\; 1 - \bigl(2\lambda \cdot (q_{-2} - q_2) - c \cdot \lambda^2 \cdot (q_{-2} + q_2)\bigr).
\end{aligned}$$

Now it suffices to show

$$\begin{aligned}
& 2\lambda \cdot (q_{-2} - q_2) + c \cdot \lambda^2 \cdot (q_{-2} + q_2) \;\geq\; \delta' \\
\Leftrightarrow \quad & c \cdot \lambda \cdot \frac{q_{-2} + q_2}{2} + \frac{\delta'}{2\lambda} \;\leq\; q_{-2} - q_2
\end{aligned}$$

Since $(q_{-2} + q_2)/2 \leq 1/2$ and $\beta \leq q_{-2} - q_2$, the next inequality implies the last inequality:

$$\frac{c\lambda}{2} + \frac{\delta'}{2\lambda} \leq \beta.$$

Now choose λ such that $0 < \lambda \leq \beta/c$. With $\delta' := \beta \cdot \lambda > 0$ the inequality is fulfilled and $\mathrm{e}^{-2\lambda} q_{-2} + \mathrm{e}^{2\lambda} q_2 \leq 1 - \delta'$ for m large enough.

The sum in the middle of (1) can be bounded the following way. Note that $r := \mathrm{e}^{2\lambda}/m^2 < 1/2$ for m large enough.

$$\begin{aligned}
\sum_{j\geq 2} \mathrm{e}^{2j\lambda} q_{2j} \;&=\; \sum_{j\geq 2} (j+1) m^{-2j} \mathrm{e}^{2j\lambda} \Big/ (1-p_0) \;=\; \sum_{j\geq 2} (j+1) \left(\frac{\mathrm{e}^{2\lambda}}{m^2}\right)^j \Big/ \Omega(m^{-2}) \\
&=\; O\Big(m^2 \cdot \sum_{j\geq 2} (j+1) \cdot r^j\Big) \;=\; O\Big(m^2 \cdot r^2 \cdot \sum_{j\geq 2} (j+1) \cdot r^{j-2}\Big) \\
&=\; O\Big(m^{-2} \cdot \sum_{j\geq 0} (j+3) \cdot r^j\Big) \;=\; O\Big(m^{-2}\Big)
\end{aligned}$$

The last equality follows from

$$\sum_{j\geq 0} (j+3) \cdot r^j = \sum_{j\geq 0} j \cdot r^j + 3 \sum_{j\geq 0} r^j = \frac{r}{(1-r)^2} + \frac{3}{1-r} = \frac{1}{1-r}\left(\frac{r}{1-r} + 3\right) \leq \frac{4}{1-r}$$

for m sufficiently large.

Finally, by our choice of $b(\ell)$, $\ell - 2i =: d$ is at least 3. Hence, the last term of (1) is

$$\frac{\mathrm{e}^{d\lambda} m^{-d}}{\Omega(m^{-2})} \;=\; O\left(\frac{\left(\mathrm{e}^{\lambda}\right)^d}{m^{d-2}}\right) \;=\; O\left(\frac{\mathrm{e}^{3\lambda}}{m} \cdot \left(\frac{\mathrm{e}^{\lambda}}{m}\right)^{d-3}\right) \;=\; O(m^{-1}).$$

Altogether, the third condition is for large ℓ fulfilled with a bound $1 - \delta$, where $\delta < \delta'$ is even a positive constant.

The last condition follows easily. We have the same sum without the constant term $\mathrm{e}^{-2\lambda} q_{-2}$. Hence, this sum is bounded above by $\mathrm{e}^{-2\lambda} + O(m^{-1})$ and by a constant, since λ is a constant.

(1+1) EA, $h = 2$. W.l.o.g. let $\ell \equiv 1 \bmod 32$. Since $h = 2$, we have $\ell = (m+1)/3$. We consider the first $2^{d \cdot m}$ steps, where $d > 0$ is an appropriate constant. Let A be the event that there is no step flipping at least $(1/8)(\ell - 1) + 1 > m/24$ bits. For a single step, the probability to flip at least $\lfloor m/24 \rfloor$ bits is at most

$$\binom{m}{\lfloor m/24 \rfloor} \left(\frac{1}{m}\right)^{\lfloor m/24 \rfloor} \leq \frac{1}{\lfloor m/24 \rfloor !} = 2^{-\log(\lfloor \frac{m}{24} \rfloor !)} = 2^{-\Omega(m \log m)}.$$

Hence, $\mathrm{Prob}(A)$ is lower bounded by $1 - 2^{d \cdot m} \cdot 2^{-\Omega(m \log m)} = 1 - 2^{-\Omega(m \log m)}$. In the following, we assume A and have to consider conditional probabilities. However, if we consider events B with $\mathrm{Prob}(B) = \Omega(1/p(m))$ where $p(m)$ is a polynomial then $\mathrm{Prob}(B \mid A) = \Omega(1/p(m))$, too. Thus we may as well work with unconditioned probabilities.

Starting with an initial path length $\ell_k = \ell$, we wait for the first point of time where the ℓ_k-value is at most $(7/8)(\ell - 1) + 1 \approx (7/8)\ell$. Then the ℓ_k-value is at least $(3/4)(\ell-1)+1 \approx (3/4)\ell$. We show that with a probability exponentially close to 1 the ℓ_k-value increases to at least $(7/8)(\ell-1)+1$ within at most $16m^3$ steps without getting short before. The augmenting path is considered *short* if its length is at most $(1/8)(\ell-1)+1$. Then it takes exponentially many of such attempts to shrink the augmenting path until the final improvement of the matching can take place. Our pessimistic assumption is that all accepted mutation steps with $j > 2$ flipping bits decrease the ℓ_k-value by j. We argue that these steps decrease the ℓ_k-value by less than $(1/16)\ell$ with a probability exponentially close to 1. Then we consider the changes of the ℓ_k-value caused by exactly two flipping bits. We show that these steps succeed in increasing the ℓ_k-value by $(1/4)(\ell-1) - \lfloor (1/16)\sqrt{\ell} \rfloor$ with a probability exponentially close to 1. This results in an ℓ_k-value larger than $(7/8)(\ell-1)+1$ within at most $16m^3$ steps.

First we account for the effect of mutation steps where more than 2 bits flip. If such a mutation does not decrease the ℓ_k-value, we ignore it. For a decreasing step where $\ell_k - \ell_{k-1} \le -2j$, $j \ge 2$, we bound the probability in the following way. It is necessary to flip at least the $2i$ leftmost edges and the $2(j-i)$ rightmost edges of the augmenting path for any $i \in \{0,\ldots,j\}$. Hence, $(j+1)(1/m)^{2j}$ is a correct upper bound. With a probability exponentially close to 1 there is no step in the phase decreasing ℓ_k by at least $\sqrt{m}/5$:

$$1 - 16m^3 \cdot \frac{(1/2)\sqrt{m}/5+1}{m^{\sqrt{m}/5}} = 1 - O(m^{-\sqrt{m}}) = 1 - 2^{-\Omega(m^{1/2}\log m)}.$$

We show that for all $2j$, $4 \le 2j < \sqrt{m}/5$, the number of steps decreasing by $2j$ is at most $\sqrt{m}/(5\cdot 2j)$ with a probability exponentially close to 1. The probability that the phase contains at least $d := \lfloor \sqrt{m}/(10j) \rfloor$ steps decreasing ℓ_k by $2j$ is upper bounded by

$$\begin{aligned}\binom{16m^3}{d}\left(\frac{j+1}{m^{2j}}\right)^d &\le \frac{16^d m^{3d}(j+1)^d}{d!\cdot m^{2j\cdot d}} \le \frac{(16+j+1)^d}{d!\cdot m^{(2j-3)d}} \le \frac{1}{d!\cdot m^{(2j-4)d}}\\ &\le \left.\begin{cases} 2^{-\log\left(\left\lfloor\frac{\sqrt{m}}{20}\right\rfloor!\right)} & \text{if } j=2\\ 2^{-(2j-4)\left\lfloor\frac{\sqrt{m}}{10j}\right\rfloor\log m} & \text{if } j\ge 3\end{cases}\right\} = 2^{-\Omega(m^{1/2}\log m)}.\end{aligned}$$

This results in a decrease of ℓ_k of at most $(1/2)(m/25) \le \ell/16$ with a probability exponentially close to 1.

At any time before the path is short, there are at least two possibilities to shorten P_k by exactly 2 edges and there are at least 2 possibilities to lengthen P_k by exactly 2 edges The probability of such a *relevant* 2-bit flip is at least $(4/m^2)(1-1/m)^{m-2} \ge 4/(\mathrm{e}m^2) > 1/m^2$. With a probability exponentially close to 1 the phase contains $8m$ relevant steps. These $8m$ relevant steps form $4m$ pairs. We call a pair *clean* if there is no other accepted mutation step in between the paired steps and estimate the number of *dirty* pairs by the number of accepted mutation steps flipping more than 2 bits. For an accepted step flipping more than 2 bits, at least 4 bits in the environment E_k (defined in the proof of Claim 4 for the (1+1) EA and $h = 2$, page 17) have to flip. This happens only with a probability of at most c/m^4 for a certain constant c and the probability that there are at most $d := \lfloor (1/120)\sqrt{m} \rfloor$ dirty pairs in the phase is at most

$$\binom{16m^3}{d}\left(\frac{c}{m^4}\right)^d \le \frac{16^d m^{3d} c^d}{d! \cdot m^{4d}} \le \frac{(16+c)^d}{d! \cdot m^d} \le 2^{-\log(d!)} = 2^{-\Omega(m^{1/2}\log m)}.$$

With a probability exponentially close to 1 we have at least $4m - \lfloor (1/120)\sqrt{m} \rfloor > (7/2)m$ clean pairs and the number of dirty pairs is at most $\lfloor (1/120)\sqrt{m} \rfloor$. These dirty pairs decrease the ℓ_k-value by at most $(1/30)\sqrt{m} < (1/16)\sqrt{\ell}$ if we pessimistically assume that each dirty pair decreases ℓ_k by 4. All steps considered so far decrease the ℓ_k-value by at most $\ell/16 + (1/16)\sqrt{\ell} < (1/8)(\ell - 1)$. Hence, we can assume for the remaining clean pairs that the initial ℓ_k-value is at least $(5/8)(\ell - 1) + 1$. A clean pair is called *relevant pair* if either both steps increase or decrease ℓ_k. We investigate the probabilities of increasing and decreasing clean pairs. The probability that a step in a pair is increasing is always at least 1/2 and a clean pair is increasing with a probability of at least 1/4. The probability that a step in a pair is decreasing is always at least 1/3 and a clean pair is decreasing with a probability of at least 1/9. Thus the probability of a relevant pair is at least 13/36 and with a probability exponentially close to 1 there are at least m relevant pairs among the $(7/2)m$ clean pairs. For an upper bound on the probability of a decreasing clean pair we pessimistically assume that the path is in marginal position in the first step, i. e., one endpoint is in column 0 or ℓ. With a probability of 1/4 the first step decreases the length *and* preserves the marginal position. With a probability of 1/4 the first step decreases the length such that thereafter the path is not in marginal position. In the first case, the next step is decreasing with a probability of 1/2 and in the second case the next step is decreasing with a probability of 1/3. Thus $1/4 \cdot 1/2 + 1/4 \cdot 1/3 = 5/24$ is an upper bound on the probability of a decreasing pair. Now we pessimistically assume that the probability of an increasing pair is 1/4 and the probability of a decreasing pair is 5/24. We map some ℓ_k-values to b-values in the following way: $\ell_k \ge (\ell - 1)/4 + 1$ with $(\ell_k - 1) \equiv 0 \bmod 4$ is mapped to $((\ell_k - 1) - (1/8)(\ell - 1))/4$. Thus $\ell_k = (1/8)(\ell - 1) + 1$ corresponds to the b-value 0, the initial path length $(5/8)(\ell - 1) + 1$ corresponds to $(2/16)(\ell - 1)$, and the target length $(7/8)(\ell - 1) + 1$ corresponds to $(3/16)(\ell - 1)$. Note that b-values are integral numbers $0, \ldots, (3/16)(\ell - 1)$. The relevant pairs correspond to the outcomes of the coin flips in a coin-tossing game where Alice's initial capital is $A = (2/16)(\ell - 1)$, Bob's is $(1/16)(\ell - 1)$. Alice wins a round with a probability of $p = 6/11$. Alice wins the game with a probability of $1 - (5/6)^{(2/16)\ell}(1 - (5/6)^{(2/16)\ell})/(1 - (5/6)^{(3/16)\ell}) \ge 1 - (5/6)^{(1/8)\ell} = 1 - 2^{-\Omega(\ell)}$. This implies that the augmenting path is never shorter than $(1/8)(\ell - 1) + 1$. We show that the game is indeed finished within the restricted time of a phase with high probability. With a probability exponentially close to 1 we have m relevant pairs and a relevant pair is increasing with a probability of at least 6/11. The probability of at least $(23/44)m$ increasing pairs is exponentially close to 1. This is a surplus of $(2/44)m$ more increasing than decreasing pairs and implies that the path has grown by $(8/44)m > (1/4)\ell$ to length at least $(7/8)(\ell - 1) + 1$ and thereby finished the coin-tossing game. With this assumption, the probability of winning the coin tossing game only increases. $\square$

We have seen that our results are not difficult in the case of the local (1+1) EA. Only the general (1+1) EA can escape eventually from each local optimum. The probability of flipping many bits in one step is essential and makes the analysis difficult. The result of Theorem 4 has the drawback of stating a lower bound on the expected optimization time and not a an exponential lower bound which holds with a probability exponentially close to 1. This offers the chance that a multistart strategy may have a polynomially bounded expected optimization time. However, many of our results hold already with a probability exponentially close to 1. The missing link is that we obtain a non-perfect matching with $n/2 - O(1)$ edges and one augmenting path of length $\Omega(\ell^{\varepsilon})$ with large probability. It is already difficult to derive properties of the first matching created by the algorithms.

Conclusions

Evolutionary algorithms without problem-specific modules are analyzed for the maximum matching problem. The results show how heuristics can "use" algorithmic ideas not known to the designer of the algorithm. Moreover, this is one of the first results where an EA is analyzed on a well-known combinatorial problem.

A The Implementation of the (1+1) EA

W. l. o. g. we assume that the graph has no isolated vertices such that $n = O(m)$, otherwise the set of vertices incident to at least one edge can be determined in time $O(m)$.

The random first search point a can be produced and evaluated in time $O(m)$. We compute the following parameters:

- $s := a_1 + \cdots + a_m$, the number of chosen edges,
- $d := (d_1, \ldots, d_n)$, the degree vector where d_i is the number of chosen edges which have i as vertex,
- $c := \sum_{1 \le i \le n} \binom{d_i}{2}$, the collision number.

Then $f(a) = -c$, if $c > 0$, and $f(a) = s$ otherwise. Moreover, we compute the probabilities b_i, the probability of exactly i flipping bits in a mutation step. Obviously, $b_i = \binom{m}{i}(\frac{1}{m})^i(1 - \frac{1}{m})^{m-i}$ and can be computed in time $O(1)$ if b_{i-1} has been computed.

For the implementation of the loop we use a random number $r \in [0, 1]$. By linear search we find j such that $b_{j-1} \le r < b_j$ (where $b_{-1} = 0$). This takes expected time $O(1)$, since the expected number of flipping bits is 1. In order to choose randomly j flipping bits, we choose the positions randomly in $\{1, \ldots, m\}$. If some position is chosen repeatedly, this choice has to be repeated. As long as $j \le m/2$, the expected number of repetitions to find the next flipping position is bounded by two. Otherwise, it is bounded by m. The expected time equals $O(1)$, since the probability of more than $m/2$ flipping bits is exponentially small. The parameters s, d, and c can be updated within the same time bound.

Obviously, we can proceed in a similar way for the local (1+1) EA.

B Maximum Matchings Can Be Computed in Polynomial Time in the Black-box Scenario

In the classical algorithmic scenario, a matching algorithm gets an explicit representation of the graph $G = (V, E)$ as input. In the black-box scenario, algorithms can only evaluate a fitness function. Initially, an algorithm has no knowledge of the graph's structure, except that there are m edges. It gains information about the structure by evaluating the fitness function for search points $a \in \{0, 1\}^m$. These fitness evaluations can be viewed as queries to an oracle. Then the black-box complexity, introduced in Droste, Jansen, Tinnefeld, and Wegener (2002), is the expected number of queries the best (randomized) algorithm needs in the worst case. In this appendix we show that the matching problem has polynomial black-box complexity and, furthermore, that it can be solved in polynomial time in this scenario. We assume that the fitness function has the following properties:

i) All matchings have higher fitness values than every non-matching,
ii) all matchings of the same size share the same fitness value,
iii) if both search points a and b encode matchings and a encodes a smaller matching than b, then $f(a) < f(b)$.

Theorem 6. *The black-box complexity of the maximum matching problem is at most* $\binom{m}{2} + 2 = O(m^2)$.

Proof. We make use of a deterministic search strategy. First of all, we query $a = (0, \ldots, 0)$ and obtain the fitness value t of the empty matching. Then, we query all subsets of E consisting of two edges, i.e., we query all $\binom{m}{2}$ possible bit strings with exactly two 1-bits in it and utilize t as a threshold; non-matchings have fitness values smaller than t and matchings of size 2 have all the same fitness value greater than t. Thereby, we learn for all pairs of edges whether they have one endpoint in common. This information is kept in a table of size $\Theta(m^2)$.

Now, given an arbitrary subset M of E, we can decide (in polynomial time) whether M is a matching. We only have to look up at most $\binom{|M|}{2} \leq \binom{m}{2}$ table entries for all pairs of edges in M. By enumerating all 2^m subsets of E we can determine all maximum matchings. Finally, we query a bit string which is the encoding of some maximum matching. □

This proves the theorem. Nonetheless, in the second part of the black-box algorithm, exponential computational effort is required, which the algorithm is not accounted for in the black-box scenario. But we can reduce the time complexity of the second part to polynomial time in the following way. The constructed table is effectively the adjacency matrix of the *line graph* of G, denoted $L(G)$. The vertices of $L(G)$ are the edges (sometimes called *lines*) of G, with two vertices of $L(G)$ adjacent whenever the corresponding edges of G are. So $L(G)$ has m nodes and at most $\binom{m}{2}$ edges. G is also called *root graph* of $L(G)$. Note that not all graphs have a root graph, i.e., not all graphs are line graphs. In fact, the complete graph K_3 has two root graphs, namely K_3 and the complete bipartite graph $K_{1,3}$. Fortunately, all other connected line graphs have unique root graphs (e.g., Harary (1969)). It is possible to reconstruct the root graph G from its line graph $L(G)$, unless $L(G) = K_3$. Due to Lehot (1974), given that the input $L(G) \neq K_3$ indeed is a line graph, the computation of its root graph G requires time $O(N)$ where N is the number of vertices of $L(G)$.

We do not assume that G is connected and distinguish two types of components of G. Components consisting of only one node contain no edges. While computing a maximum matching, we can ignore these components completely. They are not represented in the line graph $L(G)$ anyway. We name the remaining components consisting of at least two nodes and at least one edge *proper components*. Each component of $L(G)$ corresponds to a proper component of G. So computing a maximum matching from the table can be outlined like this:

1) Partition the graph $L(G)$ into components by a DFS on $L(G)$.
2) Consider each component of $L(G)$ separately, i.e., compute a maximum matching on the corresponding proper component of G.
   a) If a component of $L(G)$ consists of at most three nodes, solve the problem by "brute force." Therefore, enumerate all edge subsets of the corresponding component in G. Employ the table to decide whether a subset is a matching and choose a maximum matching.

b) Otherwise, the considered component of $L(G)$ is not K_3. Use Lehot's algorithm and obtain the root graph of the component in time linear in the number of nodes of the component in $L(G)$. Apply any polynomial time maximum matching algorithm to the root graph of the component.

3) Compose a maximum matching for G from the maximum matchings of its (proper) components.

Obviously, the overall time is a polynomial in m.

C $O(m^\ell)$ Bound for the local (1+1) EA

We have assumed that there always exists an augmenting path P of length at most ℓ, ℓ odd. Partition the series of mutation steps into phases of length $\lceil \ell/2 \rceil$. The probability to flip the edges of P from left to right in the next phase by $\lfloor \ell/2 \rfloor$ 2-bit flips and a final 1-bit flip is at least

$$\left(\frac{1}{2} \cdot \frac{2}{m(m-1)}\right)^{\lfloor \ell/2 \rfloor} \cdot \frac{1}{2} \cdot \frac{1}{m} \;=\; \Omega\big(1/m^\ell\big).$$

That means, we determine the next $\ell' \le \lceil \ell/2 \rfloor$ mutation steps that lead to a successful attempt and consider all other series of the next ℓ' mutations as error. By our assumption, we can pick a new path P and investigate the next $\ell'' \le \lceil \ell/2 \rceil$ steps if the previous phase was unsuccessful. The number of phases is a random variable $T = \min\{i \mid i\text{th phase is successful}\}$ with expectation $\mathrm{E}(T) = O(m^\ell)$. This leads to the bound $O(\ell m^\ell)$ since the length of a phase is $O(\ell)$. We stick to this bound for $\ell = 1$.

For $\ell \ge 3$, we now finish a phase prematurely after the first mutation step that differs from the determined sequence of events in a successful phase and start over. So each phase has a random length X_i. In each step, the probability to finish the phase is at least $1/2$. (In fact it is at least $1-1/(2m)$.) All X_i are stochastically dominated by a random variable Y following the geometric distribution with parameter $1/2$ and, therefore, $\mathrm{E}(Y) = 2$. We are interested in the expected total length of T phases, where the first $T-1$ phases are unsuccessful.

$$\begin{aligned}
\mathrm{E}(X_1 + \cdots + X_T) \;&=\; \sum_{1 \le t < \infty} \mathrm{E}(X_1 + \cdots + X_T \mid T = t) \cdot \mathrm{Prob}(T = t) \\
&= \sum_{1 \le t < \infty} \Big(\sum_{1 \le i \le t-1} \mathrm{E}(X_i \mid T = t) + E(X_T \mid T = t) \Big) \cdot \mathrm{Prob}(T = t)
\end{aligned}$$

The second expectation is bounded by $\lceil \ell/2 \rceil$. For $i < t$,

$$\begin{aligned}
&\mathrm{E}(X_i \mid T = t) \;\le\; E(X_i \mid X_i < \lceil \ell/2 \rceil) \\
&= \sum_{1 \le k \le \lceil \ell/2 \rceil} k \cdot \mathrm{Prob}(X_i = k \mid X_i < \lceil \ell/2 \rceil) \;=\; \sum_{1 \le k \le \lceil \ell/2 \rceil} k \cdot \frac{\mathrm{Prob}(X_i = k)}{\mathrm{Prob}(X_i < \lceil \ell/2 \rceil)} \\
&\qquad \le\; 2 \sum_{1 \le k \le \lceil \ell/2 \rceil} k \cdot \mathrm{Prob}(X_i = k) \;\le\; 2 \cdot \mathrm{E}(X_i) \;\le\; 2 \cdot \mathrm{E}(Y) \;\le\; 4.
\end{aligned}$$

Now,

$$\begin{aligned}\mathrm{E}(X_1 + \cdots + X_{T-1} + X_T) \;&\le\; \sum_{1\le t<\infty} \bigl((t-1)\cdot 4 + \ell\bigr)\cdot \mathrm{Prob}(T=t)\\ &\le\; \ell + 4\cdot \sum_{1\le t<\infty} t\cdot \mathrm{Prob}(T=t) \;=\; \ell + 4\mathrm{E}(T) \;=\; O(\ell + m^{\ell}) \;=\; O(m^{\ell}).\end{aligned}$$

References

Blum, N. (1999). A simplified realization of the Hopcroft-Karp approach to maximum matching in general graphs. Technical report, Universität Bonn.

Droste, S., Jansen, T., Tinnefeld, K., and Wegener, I. (2002). A new framework for the valuation of algorithms for black-box optimization. *Proc. of the 7th Foundations of Genetic Algorithms Workshop (FOGA VII)*, 197–214. To appear in 2003.

Droste, S., Jansen, T., and Wegener, I. (1998). On the optimization of unimodal functions with the (1+1) evolutionary algorithm. *Proc. of the 5th Conf. on Parallel Problem Solving from Nature (PPSN V),* LNCS 1498, 13–22.

Droste, S., Jansen, T., and Wegener, I. (2002). On the analysis of the (1+1) evolutionary algorithm. Theoretical Computer Science 276, 51–82.

Feller, W. (1971). *An Introduction to Probability Theory and Its Applications.* Wiley.

Hajek, B. (1982). Hitting-time and occupation-time bounds implied by drift analysis with applications. Advances in Applied Probability 14, 502–525.

Harary, F. (1969). *Graph Theory.* Addison Wesley.

He, J. and Yao, X. (2001). Drift analysis and average time complexity of evolutionary algorithms. Artificial Intelligence 127, 57–85.

Hopcroft, J. E. and Karp, R. M. (1973). An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. SIAM Journal on Computing 2, 225–231.

Jansen, T. and Wegener, I. (2001). Real royal road functions – where crossover provably is essential. *Proc. of the 3rd Genetic and Evolutionary Computation Conference (GECCO 2001)*, 375–382.

Jansen, T. and Wegener, I. (2002). The analysis of evolutionary algorithms – a proof that crossover really can help. Algorithmica 34, 47–66.

Lehot, P. (1974). An optimal algorithm to detect a line graph and output its root graph. Journal of the ACM 21, 569–575.

Micali, S. and Vazirani, V. V. (1980). An $O(\sqrt{|V|}\cdot|E|)$ algorithm for finding maximum matching in general graphs. *Proc. 21st Annual Symp. on Foundations of Computer Science (FOCS)*, 17–27.

Sasaki, G. H. and Hajek, B. (1988). The time complexity of maximum matching by simulated annealing. Journal of the ACM 35, 387–403.

Scharnow, J., Tinnefeld, K., and Wegener, I. (2002). Fitness landscapes based on sorting and shortest paths problems. *Proc. of the 7th Conf. on Parallel Problem Solving from Nature (PPSN VII),* LNCS 2439, 54–63.

Vazirani, V. V. (1994). A theory of alternating paths and blossoms for proving correctness of the $O(\sqrt{V}E)$ maximum matching algorithm. Combinatorica 14, 71–109.

Wegener, I. (2001). Theoretical aspects of evolutionary algorithms. *Proc. of the 28th Internat. Colloq. on Automata, Languages and Programming (ICALP),* LNCS 2076, 64–78.

Wegener, I. and Witt, C. (2002). On the analysis of a simple evolutionary algorithm on quadratic pseudo-boolean functions. Journal of Discrete Algorithms. To appear.